

## Assignment - 16

### Task 1

Write a simple program to show inheritance in scala.

### Task 2

Write a simple program to show multiple inheritance in scala

### Task 3

Write a partial function to add three numbers in which one number is constant and two numbers can be passed as inputs and define another method which can take the partial function as input and squares the result.

### Task 4

Write a program to print the prices of 4 courses of Acadgild:

Android App Development -14,999 INR

Data Science - 49,999 INR

Big Data Hadoop & Spark Developer – 24,999 INR

Blockchain Certification – 49,999 INR

using match and add a default condition if the user enters any other course.

### Task 1

Write a simple program to show inheritance in scala.

```
package Assignment15_1

class Superclass // Super or parent class, going to be extended by base class
{
    val value1:String = "Assignment 15.1 example code"
}
class baseclass extends Superclass{ // base or derived class extends parent class
    val value2:String = "Scala Single Inheritance"

    println("value1="+ value1)
    println("value2="+ value2)
}
object Main{
    def main(args: Array[String]): Unit={
        new baseclass()
    }
}
```

## Output

The screenshot shows an IDE with two tabs: `Employee.scala` and `Superclass.scala`. The `Superclass.scala` tab is active, displaying the following code:

```
1 package Assignment15_1
2
3 class Superclass // Super or parent class, going to be extended by base class
4 {
5     val value1:String = "Assignment 15.1 example code"
6 }
7 class baseclass extends Superclass{ // base or derived class extends parent class
8     val value2:String = "Scala Single Inheritance"
9
10    println("value1="+ value1)
11    println("value2="+ value2)
12 }
13 object Main{
14     def main(args: Array[String]): Unit={
15         new baseclass()
16     }
17 }
```

The `Employee.scala` tab is also visible, showing a package declaration: `package Assignment15_1`.

Below the code editor, the `Run` tab is active, showing the output of the program:

```
Run Main
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
value1=Assignment 15.1 example code
value2=Scala Single Inheritance
Process finished with exit code 0
```

## Task 2

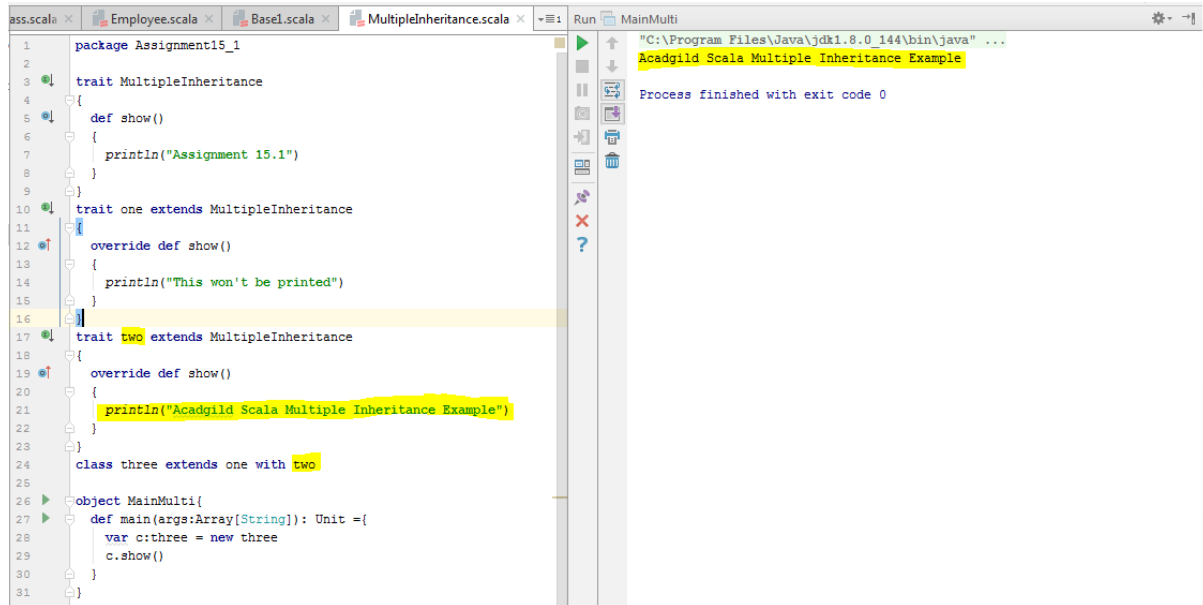
Write a simple program to show multiple inheritance in scala

```
package Assignment15_1

trait MultipleInheritance //parent trait
{
    def show() // defining the function show()
    {
        println("Assignment 15.1")
    }
}
trait one extends MultipleInheritance // extending the parent trait
{
    override def show()
    {
        println("This won't be printed")
    }
}
trait two extends MultipleInheritance // extending the parent trait
{
    override def show()
    {
        println("Acadgild Scala Multiple Inheritance Example")
    }
}
class three extends one with two //extending the base traits, calling the function
show()

object MainMulti{
    def main(args:Array[String]): Unit ={
        var c:three = new three // it will call last function which is mentioned in the
class three, changing the order will give different result
        c.show()
    }
}
```

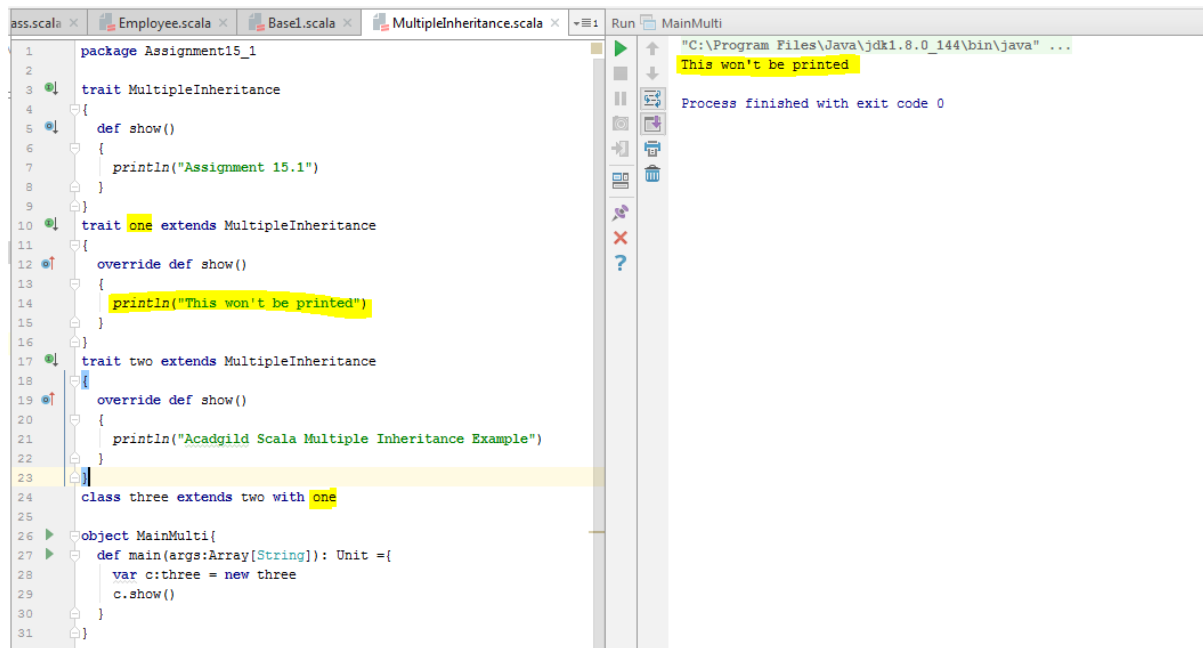
## Output



```
1 package Assignment15_1
2
3 trait MultipleInheritance
4 {
5   def show()
6   {
7     println("Assignment 15.1")
8   }
9 }
10 trait one extends MultipleInheritance
11 {
12   override def show()
13   {
14     println("This won't be printed")
15   }
16 }
17 trait two extends MultipleInheritance
18 {
19   override def show()
20   {
21     println("Acadgild Scala Multiple Inheritance Example")
22   }
23 }
24 class three extends one with two
25
26 object MainMulti{
27   def main(args:Array[String]): Unit ={
28     var c:three = new three
29     c.show()
30   }
31 }
```

Run MainMulti

"C:\Program Files\Java\jdk1.8.0\_144\bin\java" ...  
Acadgild Scala Multiple Inheritance Example  
Process finished with exit code 0



```
1 package Assignment15_1
2
3 trait MultipleInheritance
4 {
5   def show()
6   {
7     println("Assignment 15.1")
8   }
9 }
10 trait one extends MultipleInheritance
11 {
12   override def show()
13   {
14     println("This won't be printed")
15   }
16 }
17 trait two extends MultipleInheritance
18 {
19   override def show()
20   {
21     println("Acadgild Scala Multiple Inheritance Example")
22   }
23 }
24 class three extends two with one
25
26 object MainMulti{
27   def main(args:Array[String]): Unit ={
28     var c:three = new three
29     c.show()
30   }
31 }
```

Run MainMulti

"C:\Program Files\Java\jdk1.8.0\_144\bin\java" ...  
This won't be printed  
Process finished with exit code 0

## Task 3

Write a partial function to add three numbers in which one number is constant and two numbers can be passed as inputs and define another method which can take the partial function as input and squares the result.

```
package Assignment15_2

class PartialClass
{
    def squareFunc(x: Int): Unit = {
        println("Squares = " + x*x) // defined a function to square the input's
    }

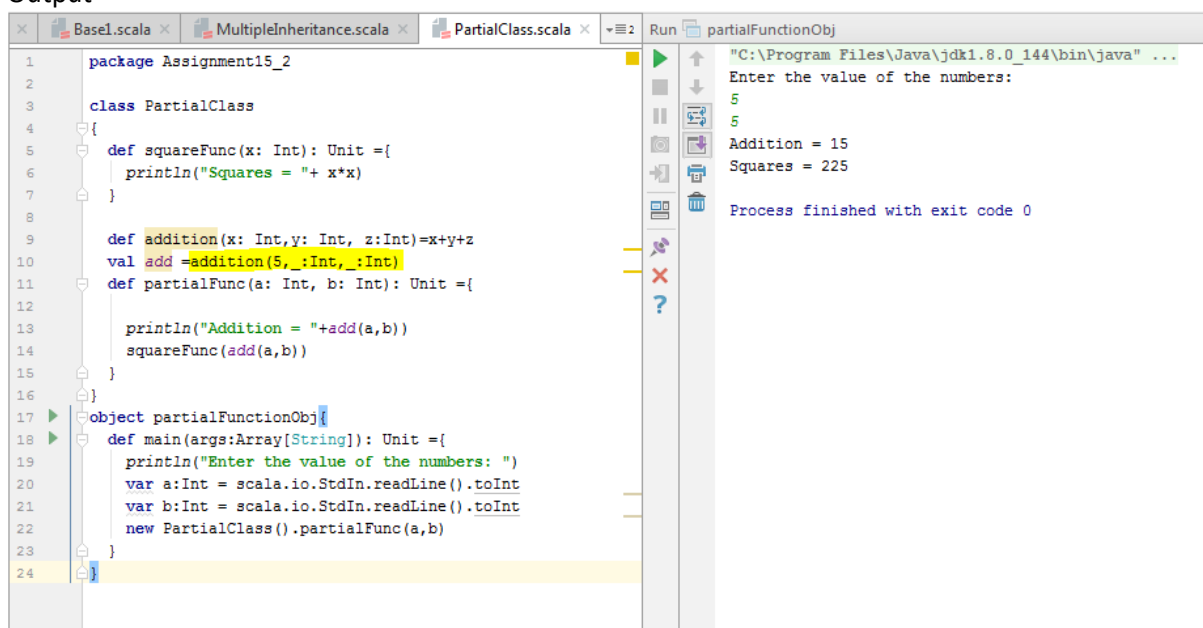
    def addition(x: Int, y: Int, z: Int) = x + y + z // a function to add constant + value1 + value2
    val add = addition(5, _: Int, _: Int) // the constant value = 5
    def partialFunc(a: Int, b: Int): Unit = { // another method to define a value for
        constant

        println("Addition = " + add(a, b))
        squareFunc(add(a, b))
    }
}

object partialFunctionObj { // singleton object to call the functions
    def main(args: Array[String]): Unit = {
        println("Enter the value of the numbers: ")
        var a: Int = scala.io.StdIn.readLine().toInt // reading the input value
        var b: Int = scala.io.StdIn.readLine().toInt
        new PartialClass().partialFunc(a, b) //
    }
}
```

Here the constant is x and we defined the value of x as 5, we have two variables a and b, we pass a=y=5 and b=z=5, we get the  $x+y+z = 5+5+5 = 15$ .  
15 is the output of the partial function is squared  $15*15$  in the **squareFunc** which is 225.

### Output



```
1 package Assignment15_2
2
3 class PartialClass
4 {
5     def squareFunc(x: Int): Unit = {
6         println("Squares = " + x*x)
7     }
8
9     def addition(x: Int, y: Int, z: Int) = x + y + z
10    val add = addition(5, _: Int, _: Int)
11    def partialFunc(a: Int, b: Int): Unit = {
12
13        println("Addition = " + add(a, b))
14        squareFunc(add(a, b))
15    }
16 }
17
18 object partialFunctionObj {
19     def main(args: Array[String]): Unit = {
20         println("Enter the value of the numbers: ")
21         var a: Int = scala.io.StdIn.readLine().toInt
22         var b: Int = scala.io.StdIn.readLine().toInt
23         new PartialClass().partialFunc(a, b)
24     }
25 }
```

Run partialFunctionObj

"C:\Program Files\Java\jdk1.8.0\_144\bin\java" ...

Enter the value of the numbers:

5

5

Addition = 15

Squares = 225

Process finished with exit code 0

## Task 4

Write a program to print the prices of 4 courses of Acadgild:

**Android App Development -14,999 INR**

**Data Science - 49,999 INR**

**Big Data Hadoop & Spark Developer – 24,999 INR**

**Blockchain Certification – 49,999 INR**

using match and add a default condition if the user enters any other course.

```
package Assignment15_2
object patternmatch
{
  def result(x: String) :String = x match
  {
    case "Android" => ("Android App Development -14,999 INR")
    case "Data Science" => ("Data Science - 49,999 INR")
    case "Big data Hadoop" => ("Big Data Hadoop & Spark Developer - 24,999 INR")
    case "Block chain" => ("Blockchain Certification - 49,999 INR")
    case => ("This course is not available")
  }
  def main(args: Array[String]): Unit =
  {
    print(result("Big Data Hadoop & Spark Developer"))
  }
}
```

### Output



```
1 package Assignment15_2
2
3 object patternmatch
4 {
5   def result(x: String) :String = x match
6   {
7     case "Android" => ("Android App Development -14,999 INR")
8     case "Data Science" => ("Data Science - 49,999 INR")
9     case "Big data Hadoop" => ("Big Data Hadoop & Spark Developer - 24,999 INR")
10    case "Block chain" => ("Blockchain Certification - 49,999 INR")
11    case => ("This course is not available")
12  }
13  def main(args: Array[String]): Unit =
14  {
15    print(result("Big Data Hadoop & Spark Developer"))
16  }
17 }
18
```

Run: "C:\Program Files\Java\jdk1.8.0\_144\bin\java" ...  
Big Data Hadoop & Spark Developer - 24,999 INR  
Process finished with exit code 0



```
1 package Assignment15_2
2
3 object patternmatch
4 {
5   def result(x: String) :String = x match
6   {
7     case "Android" => ("Android App Development -14,999 INR")
8     case "Data Science" => ("Data Science - 49,999 INR")
9     case "Big data Hadoop" => ("Big Data Hadoop & Spark Developer - 24,999 INR")
10    case "Block chain" => ("Blockchain Certification - 49,999 INR")
11    case => ("This course is not available")
12  }
13  def main(args: Array[String]): Unit =
14  {
15    print(result("Android"))
16  }
17 }
18
```

Run: "C:\Program Files\Java\jdk1.8.0\_144\bin\java" ...  
Android App Development -14,999 INR  
Process finished with exit code 0

```
1 package Assignment15_2
2
3 object patternmatch
4 {
5   def result(x: String) :String = x match
6   {
7     case "Android" => ("Android App Development -14,999 INR")
7     case "Data Science" => ("Data Science - 49,999 INR")
7     case "Big data Hadoop" => ("Big Data Hadoop & Spark Developer - 24,999 INR")
7     case "Block chain" => ("Blockchain Certification - 49,999 INR")
7     case => ("This course is not available")
8   }
9   def main(args: Array[String]): Unit =
10   {
11     print(result("Core Java"))
12   }
13 }
14
15
16
17
18
```